REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	December 1994	Professional Paper

4. TITLE AND SUBTITLE

NEURAL NETWORK CONSTRUCTION USING EVOLUTIONARY SEARCH

5. FUNDING NUMBERS

PR: ZW67
PE: 0601152N
WU: DN303002

6. AUTHOR(S)

J. R. McDonnell, W. C. Page, and D. E. Waagen

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Command, Control and Ocean Surveillance Center (NCCOSC)
RDT&E Division
San Diego, CA 92152–5001

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office and Chief of Naval Research
Independent Research Program (IR)
Arlington, VA 22217–5000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This work investigates the application of evolutionary search for training candidate hidden units in cascade-correlation learning architectures. A hybrid evolutionary search algorithm which implements techniques from evolutionary programming and evolution strategies is proposed. This approach is evaluated on selected low-dimensional examples which are non-linearly separable.

Published in *Third Annual Conference on Evolutionary Programming*, Summer 1994.

14. SUBJECT TERMS

Neural Networks
Evolutionary Programming
Signal Detection

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAME AS REPORT

21a. NAME OF RESPONSIBLE INDIVIDUAL	21b. TELEPHONE *(include Area Code)*	21c. OFFICE SYMBOL
J. McDonnell	(619) 553–5762	Code 786

NEURAL NETWORK CONSTRUCTION USING EVOLUTIONARY SEARCH

J.R. McDonnell and W.C. Page
NCCOSC, RDT&E Div
San Diego, CA 92152

D.E. Waagen
TRW Systems Integration Group
Ogden, UT 84403

ABSTRACT

This work investigates the application of evolutionary search for training candidate hidden units in cascade-correlation learning architectures. A hybrid evolutionary search algorithm which implements techniques from evolutionary programming and evolution strategies is proposed. This approach is evaluated on selected low-dimensional examples which are nonlinearly separable.

1. Introduction

By virtue of their function approximation capabilities, feedforward artificial neural networks have lent themselves well to applications in statistical pattern classification and nonlinear system mapping. Given the desired mapping (I/O) pairs (x_i,y_i), $x \in \Re^m$ and $y \in \Re^n$, a feedforward neural network can be characterized by the triple $N(W,C,F)$ which implements the mapping $f(x|W,C,F):\Re^m \rightarrow \Re^n$ where W parameterizes the network weights, C (strongly) specifies the network connectivity, and F represents the activation functions corresponding to each node. The conventional approach in most applications is to arbitrarily fix the architecture by defining the number of hidden units and the number of layers and defaulting to full connectivity between layers. A sigmoidal activation function such as the $g(u)=1/(1+e^{-u})$ normally serves as the *de facto* nonlinear mapping of each node in the network.

Given the above constraints, the standard network design approach becomes a nonlinear regression problem which seeks to determine a weight set w that minimizes an arbitrarily selected objective function $J(x,w)$. The most commonly used objective function is the sum-squared network error which satisfies the continuity requirements of gradient search methods such as backpropagation[1]. The traditional design approach can become a tedious and laborious undertaking if an acceptable objective criterion (i.e., $J < \varepsilon$) is not met due to inadequate network representation or learning.

The cascade-correlation learning architecture[2] (CCLA) has been proposed as constructive method that automates the network design process. This investigation modifies the traditional cascade-correlation training algorithm by using evolutionary

· *presented at the Third Annual Conf. on Evolutionary Programming, San Diego, 1994.*

search instead of quickprop[3] to train each candidate hidden node. At the end of each evolutionary cycle, the best 'evolved' candidate node is incorporated into the network.

The following sections discuss the CCLA and our hybrid approach for evolutionary optimization. The application of evolutionary search to the CCLA is then described. Finally, results are discussed for the parity and spiral problems.

1.1 Cascade-Correlation Architectures

Cascade-correlation architectures were introduced by Fahlman and Lebiere[2] as a means of automatically determining parsimonious network structure given a particular data set. The network is initiallized with only the input units mapped directly to the output units. New hidden units are individually incorporated into the network with input weights frozen. The input weights to each hidden unit are frozen since newly created nodes have been trained to maximize the correlation between their output and the residual output error of the network. This is the correlation aspect of the cascade-correlation architecture. A pool of hidden units can be used to increase the likelihood of finding a good candidate unit. Each new hidden unit is connected to *all* input and previously created hidden units in the network as shown in Figure 1. All of the hidden units are connected to all of the output units. After each hidden unit is incorporated into the network, additional training takes place on the weights to the output layer with the weights to the hidden units remaining fixed. Squires and Shavlik[4] have shown that faster training times and better generalization can sometimes result if all of the weights in the network are trained using backpropagation.

The hidden and output units achieve the following mappings

hidden unit:
$$x_i = g_i\left(\sum_{j=1}^{i-1} \alpha_{ij} x_j\right) \qquad m < i \le m+h \qquad (1)$$

output unit:
$$x_i = g_i\left(\sum_{j=1}^{m+h} \alpha_{ij} x_j\right) \qquad m+h < i \le m+h+n \qquad (2)$$

where h represents the number of hidden units, m the number of inputs, and n the number of outputs. The cascade-correlation architecture appears quite similar to the projection pursuit technique[5] when, assuming linear outputs, equation (2) is broken down into its linear and nonlinear components

$$x_i = a_i^T u + \sum_{j=m+1}^{m+h} \alpha_{ij} g_j\left(\alpha_j, u\right)$$

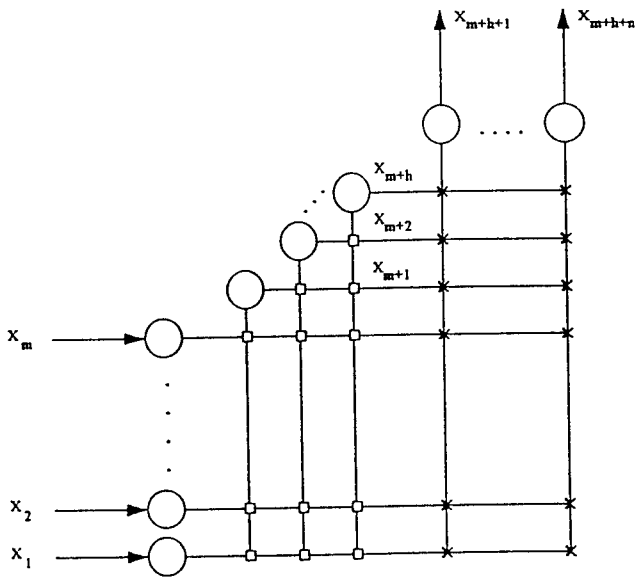

Figure 1. *The cascade-correlation network structure. The boxes indicate weights which are frozen when the unit is incorporated into the network. The crosses indicate weights which are modified after a new hidden unit is incorporated.*

where $\boldsymbol{u}=x_i \; \forall \; i \in \{1,...,m\}$ and $a_i=\alpha_{ij} \; \forall \; j \in \{1,...,m\}$. However, this view oversimplifies the repeated nonlinear transformations which actually occur as described by

$$x_i = a_i^T u + \sum_{j=m+1}^{m+h} \alpha_{ij} g_j \left(a_j^T u + \sum_{k=m+1}^{j-1} \alpha_{jk} g_k(\alpha_k, u) \right)$$

1.2 Evolutionary Search

In 1958, Brooks[6] described a *creeping random* method where k points were generated via Gaussian perturbations about a search point. The best point was kept and the process repeated. Brooks observed that "there are some rather intriguing analogies that can be made between the creeping random method and evolution." This analogy was also apparent to Fogel *et al.*[7] who proposed a random search strategy termed *evolutionary programming* (EP). Fogel *et al.*[7] proposed the following:

> *A "parent" organism is scored in terms of its ability to accomplish the desired decision making on the basis of evidence at hand. The organism is mutated to yield an "offspring" which is given the same task and scored in a similar manner. That organism which demonstrated the greatest ability to perform the required function is retained to serve as parent for a new offspring.*

More recently, D. Fogel[8,9] has refined the EP paradigm as well as applied it to a variety of problems including system identification and control. This work modifies EP by embedding two additional search strategies in the evolutionary optimization process. That is, the evolutionary search algorithm implemented in this study augments standard EP by incorporating recombination of the parents to generate offspring and by modifiying the parents before any offspring are generated. The utilization of the recombination operator produces an evolutionary search method similar to an *evolution strategy*[10] (ES). The parents are modified using the Solis and Wets[11] random optimization method. The resulting strategy has been termed a hybrid evolutionary algorithm and is given below in the nomenclature proposed by Bäck and Schwfel[10]. This hybrid approach appears to incorporate both Darwinian evolutionary learning and Lamarckian inheritance models which gives rise to the term "Lamarckian Learning." As pointed out by Davidor[12], "Larmarckism is an important model which can complement many learning algorithms."

A HYBRID-EVOLUTIONARY SEARCH ALGORITHM

$k=0;$
initialize: $P(0) = \{a_1(0),\ldots,a_\mu(0)\};$
where $a_i = (x_j, \forall j \in \{1,\ldots,n\})$
evaluate P(0): $\Phi(P(0)) = \{\Phi(a_1(0)),\ldots,\Phi(a_\mu(0))\};$
do {
modify parents: $a_i(k) = \{m,s\}_{\{SW\}}(a_i(k)) \forall i \in \{1,\ldots,\mu\}$
mutate: $a_i'(k) = m'_{\{\gamma\}}(a_i(k)) \; \forall i \in \{1,\ldots,\mu\}$
recombine: $a_i'' = r'(P(k) \cup P'(k)) \; \forall i \in \{1,\ldots,\mu\}$
evaluate P''(k): $\Phi(P''(k)) = \{\Phi(a_1''(k)),\ldots,\Phi(a_\mu''(k))\}$
select: $P(k+1) = s_{(\mu+\mu)}(P(k) \cup P''(k));$
$k=k+1;$
} while $(\iota(P(k)) \neq true)$

2. Evolving Cascaded Networks

The work conducted in this study employs evolutionary search to optimize a population of individual nodes and then the best node is brought into the network in a purely constructive manner. This contrasts other work in evolving networks which focuses on the optimization of a whole population of neural nets using both constructive and pruning mechanisms[13,14]. Benefits of the single node approach include lower computational requirements in the form of memory and processor power.

A population of candidate hidden units are randomly intialized with full connectivity as in the standard cascade-correlation learning architecture (CCLA). The

same optimization objective used to train CCLA units was employed to represent the fitness of each candidate node

$$\Phi(a_i) = \sum_o \left| \sum_p (Z_{i,p} - \overline{Z}_{i,p})(E_{p,o} - \overline{E}_o) \right|$$

where Z_p represents the output of each candidate node for pattern p and E is the residual error as measure at the networks output o. The weight vectors for each candidate unit were modified using the hybrid method given above. Mutation of each weight was accomplished using standard EP

$$w'_{i,j} = w_{i,j} + \sqrt{Sf \,/\, \Phi(a_i)} \cdot N(0,1)$$

where Sf represents the scaling factor. Recombination took place according to

$$\mathbf{w}_k = \mathbf{w}_i + \lambda(\mathbf{w}_j - \mathbf{w}_i)$$

where $i,j \in \{1,..,2N\}$, $i \neq j$, and $\lambda \sim U(0,1)$. Selection was determinstic using a $(\mu+\mu+\mu)$ strategy to choose the best μ individuals among the modified parents, the mutated offspring, and the recombined offspring. After an arbitrary number of generations, the best candidate unit was inserted into the network.

The output weights were found using a more direct approach. The inputs to the output layer of the cascade-correlation architecture for p patterns are described by

$$\mathbf{X}_p = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_m^{(1)} & x_{m+1}^{(1)} & x_{m+2}^{(1)} & \cdots & x_{m+h}^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_m^{(2)} & x_{m+1}^{(2)} & x_{m+2}^{(2)} & \cdots & x_{m+h}^{(2)} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1^{(p)} & x_2^{(p)} & \cdots & x_m^{(p)} & x_{m+1}^{(p)} & x_{m+2}^{(p)} & \cdots & x_{m+h}^{(p)} \end{bmatrix}$$

Likewise, the outputs of the network are given by

$$\mathbf{Y}_p = \begin{bmatrix} x_{m+h+1}^{(1)} & x_{m+h+2}^{(1)} & \cdots & x_{m+h+n}^{(1)} \\ x_{m+h+1}^{(2)} & x_{m+h+2}^{(2)} & \cdots & x_{m+h+n}^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m+h+1}^{(p)} & x_{m+h+2}^{(p)} & \cdots & x_{m+h+n}^{(p)} \end{bmatrix}$$

where $\mathbf{Y}_p=\mathbf{X}_p\mathbf{V}$. The optimal (in a least-squares sense) weight set $\mathbf{V}$ can be determined from $\mathbf{V}=(\mathbf{X}_p^T\mathbf{X}_p)^{-1}\mathbf{X}_p^T\mathbf{Y}_p$. Iterative deterministic methods such as the LMS rule can also be applied in determining the $n\cdot(m+h)$ output weights.

Finally, the complexity of the network was monitored using an MDL-like[15] objective function

$$J_{MDL}=\ln(\hat{\sigma}^2)+n_w\ln(p)/p$$

where n_w is the number of weights. The number of hidden units can be directly incorporated in this calculation according to

$$J_{MDL}=\ln(\hat{\sigma}^2)+\left(n\cdot(m+h)+m\cdot h+h\cdot(h-1)/2\right)\ln(p)/p$$

Ultimately, one desires to construct a network with minimal complexity cost.

3. Results

The proposed algorithm was initially tested on the 3-bit and 4-bit parity problems, respectively. Using the above formulation, solutions were found in a relatively rapid manner for a small number of evolutionary training cycles or generations (typically less than 10). Due to the discrete nature of the problem, the fitness values climbed in a precipitous fashion using evolutionary optimization with 50-100 parents. The parity problem proved insightful for tuning the algorithm in that dependence on the scaling factor was observed. For example, a Sf=10 reliably yielded architectures with only one hidden node for the 3-bit parity problem or two hidden nodes for the 4-bit parity problem. A Sf=1 often resulted in networks with more than the minimum number of nodes.

The next experiment investigated the two-spirals problem. The two-spirals problem requires the network to discriminate between two interlocking spirals which encircle the origin three times. This problem has proved troublesome for standard architectures and training methods[2]. The two-spirals are readily discriminated using the CCLA with quickprop training. Since these experiments replace the quickprop training with stochastic search similar results were expected. The training method outlined above did manage to solve the two-spiral problem, but more than the typical 10-15 hidden units were necessary. The need for additional hidden units may have resulted from a need for better local learning or increased behavioral freedom. These hypotheses are currently under investigation.

The receptive plane generated using 100 parents (without the Solis and Wets technique), *tanh* activation functions, and a $Sf=1$ are shown in Figure 2(a). The receptive plane does not appear to generalize as well the cascade-correlation classifier found using quickprop[2] in that a "sausage-link" effect occurs as opposed to a clean spiral. The number of training cycles and hidden nodes were arbitrarily limited to 300 and 24, respectively, as shown in Figure 2(b). Figure 3 compares the optimization process in terms of network

mean squared-error and complexity as each hidden node is incorporated. For this training run, Figure 3(b) shows that a flattening occurs around 10 hidden nodes before the complexity cost resumes its climb upon incorporation of additional hidden nodes.

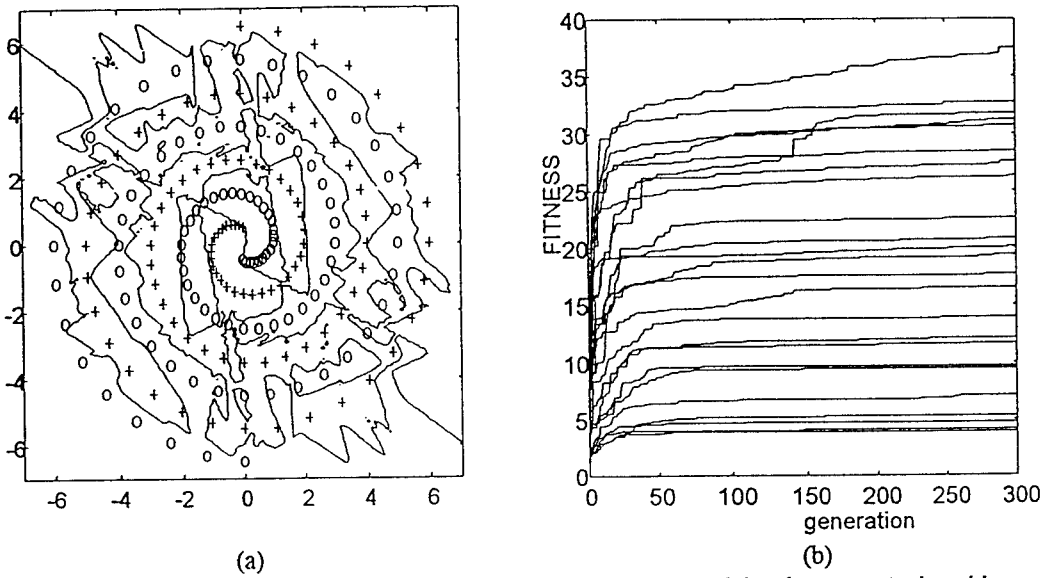

Figure 2. (a) The training points are superimposed on the contour grid for the two-spiral problem, and (b) The best fitness Φ(a) in each pool of candidate hidden units is plotted for the number of nodes added.

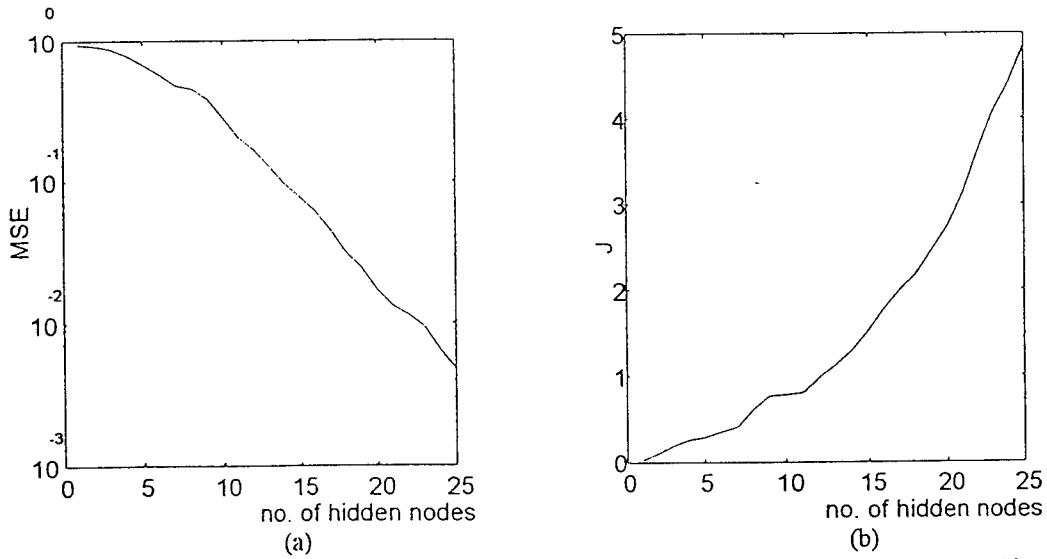

Figure 3. (a) The mean squared network error, and (b) the complexity cost J_{MDL} as new hidden nodes are incorporated into the network.

4. Conclusion

As more information and insight are gained into the dynamics of evolutionary computation, it is inevitable that components from the various search strategies (ES, EP, GA) will be combined to yield fairly robust multi-agent stochastic search techniques. This work has demonstrated the applicability of evolutionary search, albeit a hybrid approach, for use in the cascade-correlation learning architecture. More importantly, this work represents a preliminary step toward using evolutionary search in a purely constructive manner wherein limited fan-in random wired nodes[16] can be generated with a randomly chosen activation function $f \in F$. These issues will be addressed in subsequent work.

5. References

1. D.E. Rumelhart, G.E. Hinton, and R.J. Williams. "Learning internal representations by error propagation" in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, ed. D.E. Rumelhard and J.L. McClelland (MIT Press, 1986).
2. S.E. Fahlman and C. Lebiere, "The cascade-correlation learning architecture" in *Neural Information Processing Systems 2*, ed. D.S. Touretzky, (Morgan Kaufmann, 1990), p. 598.
3. S.E. Fahlman, "An empirical study of learning speed in back-propagation networks," Technical Report, CMU-CS-88-162, (1988).
4. C.S. Squires and J.W. Shavlik, "Experimental analysis of aspects of the cascade-correlation learning architecture," Machine Learning Research Group Working Paper 91-1, Comp. Sci. Dept., University of Wisconsin-Madison, (1991).
5. T.E. Flick, L.K. Jones, R.G. Priest, and C. Herman, "Pattern classification using projection pursuit," *Pattern Recognition*, Vol. 23, No. 12, (1990), pp. 1367-1376.
6. S.H. Brooks, "A discussion of random methods for seeking maxima," *Operations Research*, Vol. 6, (1958), p. 244-251.
7. L.J. Fogel, A.J. Owen, and M.J. Walsh, *Artificial Intelligence through Simulated Evolution*, (John Wiley & Sons, 1966).
8. D.B. Fogel, System Identification through Simulated Evolution: A Machine Learning Approach to Modeling, (Ginn Press, 1991).
9. D.B. Fogel, Evolving Artificial Intelligence, Ph.D. Dissertation, University of California, San Diego, (1992).
10. T. Bäck and H-P. Schwefel, "An overview of evolutionary algorithms for parameter optimization," *Evolutionary Computation*, Vol. 1, No. 1, (1993), pp. 1-24.
11. Solis, F.J. and Wets, J.B, "Minimization by random search techniques," *Mathematics of Operations Research*, 6, (1981), pp. 19-30.
12. Y. Davidor, "A genetic algorithm applied to robot trajectory generation," in *Handbook of Genetic Algorithms*, ed. L. Davis, (Van Nostrand Reinhold, 1991), pp. 144-165.
13. P.J. Angeline, G.M. Saunders, and J.B. Pollack, "An evolutionary algorithm that constructs recurrent neural networks," LAIR Technical Report 93-PA-GNARL, The Ohio State University, Columbus, Ohio, (1993).
14. J.R. McDonnell and D.E. Waagen, "Evolving neural network pattern classifiers," SPIE Vol. 2032, Proc. on Neural and Stochastic Methods in Image and Signal Processing II, San Diego, (1993).
15. M. Bello, "Enhanced training algorithms, and integrated training/architecture selection for multilayer perceptron networks," *IEEE Trans. on Neural Networks*, Vol. 3, No. 6, (1992), pp. 864-874.
16. H. Klagges and M. Soegtrop, "Limited fan-in random wired cascade-correlation," Proc. of the Third Int. Conf. on Microelectronics for Neural Networks, Edinburgh, Scotland, (1993), pp. 6-8.